Jussi Anttila

WEB-TUOTEKUVASTON HALLINTAJÄRJESTELMÄ

Tietotekniikan koulutusohjelma
Elektroniikkatuotannon suuntautumisvaihtoehto
2012

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

WEB-TUOTEKUVASTON HALLINTAJÄRJESTELMÄ

Anttila, Jussi
Satakunnan ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Kesäkuu 2012
Ohjaaja: Haataja, Rauli
Sivumäärä: 31
Liitteitä: 1

Asiasanat: JavaScript, PHP, MySQL, Ajax, käyttäjätodennus, autentikointi

---

Opinnäytetyön aiheena oli ohjelmoida netissä toimivan tuotekuvaston hallintajärjestelmä. Sen web-pohjaisen käyttöliittymän ulkoasu ja käyttötuntuma muistuttaa tavallista työpöytäsovellusta.

Järjestelmän avulla yritykset pystyvät luomaan tuotekuvaston kotisivuilleen. Käyttäjät kirjautuvat järjestelmään turvallisesti henkilökohtaisin tunnuksin. Eri tason käyttäjäryhmillä on eri tason käyttöoikeudet hallita kuvastoa.

Projekti toteutettiin käyttämällä JavaScript ja PHP -ohjelmointikieliä ja tietokantajärjestelmänä toimi MySQL. Nämä yhdistettiin keskenään Ajax-tekniikalla. Ohjelmointi suoritettiin tekstieditorilla ja PhpMyAdmin -verkkosovelluksella hallinnoitiin tietokantaa. Työ aloitettiin tyhjältä pöydältä, koska saatavilla ei ollut mitään vanhaa järjestelmää.

Tämä opinnäytetyön kirjallinen osuus selvittää työssä käytettyjen tekniikoiden perusteet ja näyttää periaatteen miten näitä tekniikoita voidaan soveltaa tämän kaltaiseen web-sivuun.

THE MANAGEMENT SYSTEM OF A WEB CATALOG

Anttila, Jussi
Satakunta University of Applied Sciences
Degree Programme in Information Technology
June 2012
Supervisor: Haataja, Rauli
Number of pages: 31
Appendices: 1

Keywords: JavaScript, PHP, MySQL, Ajax, user authentication

The purpose of this thesis was to program a management system of a catalog running on the internet. Its web based user interface's layout and using experience reminds of normal desktop application.

With the system companies can create a catalog of their products on their web site. Users log in to the system safely with their personal access codes. Different level user groups have different level privileges to administrate the catalog.

The project was carried out with JavaScript and PHP programming languages and as a database system there was MySQL. These were combined together with the Ajax technique. The programming was written with a text editor and PhpMyAdmin web application was used to manage the database. The project started from scratch because there wasn't a previous ready-made system available.

This written part of the thesis clarifies the basics of used techniques and shows the principle of how these techniques can be applied in the web page like this one.

SISÄLLYS

LYHENTEET

W3C	**W**orld **W**ide **W**eb **C**onsortium
HTML	**H**yper**t**ext **m**arkup **l**anguage
HTTP	**H**yper**t**ext **t**ransfer **p**rotocol
FTP	**F**ile **t**ransfer **p**rotocol
DHTML	**D**ynamic HTML
XHTML	E**x**tensible HTML
XML	E**x**tensible **m**arkup **l**anguage
DOM	**D**ocument **o**bject **m**odel
DTD	**D**ocument **t**ype **d**efinition
CSS	**C**ascading **s**tyle **s**heets
PHP	Rekursiivinen PHP: Hypertext Preprocessor (alkuperäinen tarkoitus oli **P**ersonal **h**ome **p**age)
MySQL	**My S**tructured **Q**uery **L**anguage
AJAX	**A**synchronous **J**avascript **a**nd **X**ML

1 JOHDANTO

Opinnäytetyön tarkoituksena oli toteuttaa yritykselle web-pohjainen hallintajärjestelmä, jolla voidaan luoda kategorioitu tuotekuvaston rakenne tietokantaan. Projekti sisältää julkisen käyttöliittymän, ylläpitoliittymän ja tietokantarakenteen toteutuksen. Oma osuuteni on toteuttaa ylläpitoliittymä ja tietokantarakenne. Yrityksen vastuulle jää julkinen liittymä, eli web-tuotekuvasto, jolla tämä ylläpito-ohjelmalla tuotettu rakenne esitetään.

Tavoitteena oli saada aikaan mahdollisimman selkeä tietokantarakenne ja ohjelmakoodi, jotta yritys itse pystyy helposti kehittämään järjestelmää ja räätälöimään sitä omien asiakkaidensa tarpeisiin.

Työympäristönä oli Unix-alusta, johon oli asennettu Apache HTTP-palvelin, MySQL -tietokantajärjestelmä ja PHP -ohjelmointialusta. Ohjelmakoodit kirjoitettiin tekstieditorilla ja tietokantaa hallinnoitiin PhpMyAdmin -verkkosovelluksella.

Tämä raportti selvittää työssä käytettyjen tekniikoiden taustoja ja perusteita sekä periaatteen, miten ne toimivat kaikki yhdessä, jotta saadaan luotua hyvin dynaaminen käyttöliittymä.

2 VAATIMUSMÄÄRITTELY

2.1 Vaatimusluettelo

Yrityksen toimesta annettiin aluksi muutamia vaatimuksia, joiden tuli täyttyä valmiissa työssä. Vaatimuksia tuli harmillisesti lisää myös työn edetessä, joten työtä oli muokattava välillä rajustikin. Seuraavana on listattuna kaikki vaatimukset.

- Käytetään vain avoimeen lähdekoodiin perustuvia tekniikoita.
- Käyttöliittymästä löytyy kaikki tarvittavat toiminnot kuvaston rakentamiseen
- Käyttöliittymän toiminta muistuttaa työpöytäsovellusta. Eli sivu ei päivity kun toimintoja tehdään, eikä siellä ole näennäisesti mitään muutakaan ”web-sivumaista” kuten linkkejä yms.
- Ohjelmakoodi ja tietokantarakenne ovat selkeitä ja loogisia, jotta yritys voi itse kehittää järjestelmää omiin ja asiakkaidensa tarpeisiin helposti.
- Ylläpidon on tapahduttava käyttäjätunnusten takana.
- Käyttäjän on vaihdettava salasanansa tietyin välein, eikä se voi olla sama kuin edellinen.
- Salasanan pitää olla vähintään tietyn mittainen ja siinä pitää olla tiettyjä merkkejä tietty vähimmäismäärä.
- Tunnukset ja salasanat arvotaan ja lähetetään oikealle henkilölle automaattisesti kun käyttäjätili lisätään, tai salasana unohtuu.
- Järjestelmässä on kolmen eri tason käyttäjiä. Eri tasoilla on toisistaan eroavat käyttöoikeudet.
    - alatason käyttöoikeudet (tuotteiden lisäys, muokkaus, poisto ja kuvien lataus)
    - keskitason käyttöoikeudet (samat kuin alatasolla, kategorointi)
    - ylätason käyttöoikeudet (samat kuin keskitasolla, käyttäjätilien lisäys, poisto ja käytön esto, salasanojen uusiminen)

Vaatimuksiin ei siis kuulunut täydelliseksi hiottu ulkoasu eikä monimutkaiset toiminnot, vaan looginen perustekniikka, jolla järjestelmä toimii ja kaikki vaatimukset täyttyvät.

2.2 Tekniikoiden valinta

Aluksi valittiin kaikki työssä käytettävät tekniikat niin, että kaikki tilaajan vaatimukset työlle voidaan täyttää. Valinnat tehtiin ensisijaisesti sen mukaan mitä palveluja oli valmiina asennettuna ja toisaalta mitkä niistä ovat tutuimpia. Tässä luvussa on lyhyesti kerrottu miksi tietty tekniikka valittiin.

Unix-palvelinkoneelle oli valmiiksi asennettu Apache HTTP-palvelinohjelmisto ja erilaisia tietokantajärjestelmiä sekä palvelinpään ohjelmointikieliä, mm. Perl ja PHP.

Tietokannaksi valittiin MySQL, koska se on nopea laajojen tietokantojen kanssa ja tuotekuvaston datamäärä voi mahdollisesti paisua suureksi. Tämä tietokanta sopii samalla myös käyttäjätilien taltioimiseen. Se on myös itselleni jonkin verran tuttu.

Palvelinpuolen ohjelmointikieleksi valittiin PHP, koska olen käyttänyt sitä MySQL:n kanssa aikaisemminkin ja muut vaihtoehdot olivat melkein tuntemattomia. PHP:lla voidaan käsitellä erilaisia tiedostoformaatteja, kuten kuvatiedostoja tai erilaisia dokumentteja. Sillä voitaisiin tarvittaessa luoda käyttäjälle tuote-esitteitä PDF -dokumentteina nettikuvaston tuoteselosteista. PHP:lla pystytään myös hoitamaan kaikki tekniset toiminnot mitä työssä tarvitaan.

Jotta hallintasivu saadaan tuntumaan työpöytäsovellukselta, otetaan JavaScript mukaan. Se lisää dynaamisuutta merkittävästi web-sivulle. JavaScript yhdistetään PHP:n kautta tietokantaan Ajax-tekniikalla. Se mahdollistaa tiedonsiirron taustalla käyttäjän havaitsematta ja toimintojen aiheuttamat muutokset näkyvät heti käyttöliittymässä ilman sivun päivittymistä.

3 KÄYTETYT TEKNIIKAT

Työssä on käytetty muutamaa eri web-sovelluskehityksen tekniikkaa. Mukana ovat HTML ja CSS sivun ulkoasua varten. JavaScript luo tarvittavan dynaamisuuden PHP:n kanssa. PHP ja MySQL keskenään hoitavat tietokantaliikenteen. Lisäksi PHP hoitaa kaikki ”älyä” vaativat toimenpiteet kuten mm. käyttäjätunnistuksen, salasanojen arpomisen ja sähköpostien lähettelyn käyttäjille. Tekniikoiden perusteita selvitetään enimmäkseen niiltä osin mitä tämän työn kannalta on tärkeää tietää.

3.1 HTML

HTML -merkkauskieli on avoimesti standardoitu kuvauskieli, jolla voidaan kuvata hypertekstiä eli hyperlinkkejä sisältävää tekstiä. Sillä voidaan merkitä esimerkiksi tekstin rakenne, kuten eritellä otsikot ja leipäteksti. Lisäksi sillä merkitään myös muita elementtejä ja objekteja, kuten kuvia ja mediaa. HTML on pääasiallinen websivujen luomiseen tarkoitettu kieli. (Wikipedian www-sivut 2012)

HTML -sivu rakentuu aina samoin. Se koostuu kahdesta osasta, jotka ovat HEAD ja BODY, eli otsikkotiedot ja sivun runko. Seuraavassa listauksessa on sen perusrakenne.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
  <!-- Tässä osiossa esitellään sivun otsikko, tyylit, JavaScript koodit yms. -->
</HEAD>

<BODY>
  <!-- Body-osioon tulee kaikki sivun sisältö -->
</BODY>
</HTML>
```

Listaus 3.1-1. Standardin mukainen HTML -sivun rakenne.

Lisätietoa HTML -sivujen rakentamisesta saa alan kirjoista ja netti on täynnä esimerkkejä. Yksityiskohtaista tietoa löytyy W3C:n (World Wide Web Consortium) kotisivuilta. W3C on järjestö joka ylläpitää mm. HTML -standardia.

3.2 CSS

CSS on HTML -koodille kehitetty tyyliohje. CSS määrittelee tyyliohjeet esimerkiksi luokittain tai yksittäisille HTML -elementeille. Nämä määrittelyt ehdottavat selaimelle kuinka web-sivu tulisi esittää. W3C ylläpitää myös CSS standardia ja järjestön kotisivuilta löytyy yksityiskohtaista tietoa tyylimäärittelyyn. (Web-opas www-sivut 2012)

Eri selaimissa tyylien oletusarvot ovat hieman erilaiset keskenään, joten standardeja tyylimäärittelyjä käyttämällä eri selaimet voivat näyttää jotain eri tavoin standardoinnista huolimatta. Tarkka CSS -tyylimäärittely HTML -sivun otsikko-osiossa kuitenkin kumoaa joka selaimen oletusmäärittelyt.

On olemassa myös selainkohtaisia tyylien lisämäärittelyjä, jotka eivät ole standardeja. Jos esimerkiksi käytetään jotain Mozillan omaa vain sille tarkoitettua tyylimäärittelyä, Internet Explorerilla sivu näyttää siltä kuin kyseistä tyylimäärittelyä ei olisi. Tällaisia määrittelyjä tulee tietysti välttää, koska sivujen halutaan yleensä toimivan samoin mahdollisimman monella selaimella.

```
h1 {            /* asettaa kaikkien H1 otsikoiden taustavärin siniseksi */
  background-color: blue;
}

.mainlink {    /* asettaa kaikkien "mainlink"-luokkaisten elementtien taustavärin siniseksi */
  background-color: blue;
}

#osio4 {       /* asettaa kaikkien elementtien taustavärin siniseksi, joiden ID-tunniste on "osio4" */
  background-color: blue;
}
```

Listaus 3.2-1. CSS -tyylimäärittelyesimerkki.

Tyyli voidaan määritellä myös suoraan web-sivun HTML -elementteihin, mutta se ei ole kovin järkevää, jos sivuilla on ryhmä elementtejä, joille halutaan samanlainen tyyli. Kun sellaisessa tapauksessa kaikkien kyseisten elementtien tyylit pitäisi vaihtaa, jouduttaisiin jokaisen elementin määrittelyt vaihtamaan erikseen. Elementtiin suoraan liitetty tyylimäärittely kumoaa otsikko-osiossa esitellyt tyylimäärittelyt, määriteltiinpä ne sinne suoraan tai luetaan erillisestä tiedostosta.

3.3 JavaScript

Tämä web-ympäristössä käytettävä komentosarjakieli on ensisijaisesti tarkoitettu mahdollistamaan dynaamista toiminnallisuutta web-sivuille. JavaScript on selaimella ajettava merkkauskieli samoin kuin HTML. Koska JavaScript toimii vain selaimen sisällä, se ei voi keskustella ulkoisen palvelimen tai tietokannan kanssa itsenäisesti. Hyödyllisyytenä tulee kuitenkin verkkoliikenteen minimaalisuus. JavaScriptin lähdekoodi – oli se sitten HTML -koodiin sisäistettynä tai erillisessä tiedostossa – on aina luettavissa asiakaspäässä. (Wikipedian www-sivut 2012)

JavaScriptillä voidaan antaa ”älyä” web-sivulle. Otetaan esimerkiksi lomake, joka tarkistaa käyttäjän antamia syötteitä ja ilmoittaa käyttäjälle heti jos jokin syötteistä on virheellinen. Näin lomakkeen täytössä säästyy aikaa ja resursseja molemmin puolin kun käyttäjä voi välittömästi korjata virheensä ilman lomakkeen lähetystä ensin palvelimelle tarkistukseen. Erilaisena esimerkkinä mainittakoon valikot, joista ensimmäisestä valitaan valtio ja seuraavasta kaupunki. Kun valtio on valittu, JavaScript muokkaa web-sivun sisältöä niin, että seuraavasta valikosta löytyvät ainoastaan valitun valtion kaupungit. Myös erilaiset visuaaliset efektit web-sivuilla on toteutettavissa JavaScriptillä. Monimutkaisten visuaalisten efektien tuottaminen JavaScriptillä on kuitenkin paljon työläämpää kuin tehdä Flash -sovellus ja upottaa se web-sivulle. Flash -sovellus tarvitsee kuitenkin erikseen asennettavan liitännäisen toimiakseen.

Turvallisuussyistä JavaScriptille on asetettu omat estonsa. Evästeitä lukuun ottamatta JavaScript ei pysty käsittelemään asiakaskoneen tiedostoja, eikä se salli tiedostojen luomista myöskään palvelimelle. Se ei pysty sulkemaan selaimen ikkunoita, joita se ei ole itse avannut, eikä voi tietää mitä muita sivuja käyttäjällä on samaan aikaan auki.

Virheellisesti usein ajatellaan, että JavaScript olisi sama kuin Java tai jollain tavalla sukua sille. On totta, että niiden molempien syntaksi muistuttaa C-kieltä ja molemmat ovat oliopohjaisia ohjelmointikieliä. Mutta JavaScriptin oliomalli perustuu prototyyppeihin, ei luokkiin kuten Javassa tai C++:ssa. Java on perusohjelmointikieli, kuten C tai C++. Niillä voidaan luoda itsenäisiä tietokoneohjelmia tai ohjelmoida kulutuselektroniikan laitteita. JavaScript on puolestaan tarkoitettu web-sovelluksiin.

Nimet ovat samankaltaisia, koska Netscapen LiveScript -niminen skriptikieli vaihdettiin Javan suosion kasvaessa JavaScriptiksi markkinointisyistä. (Negrino & Smith 2007, 3-5)

Lopuksi havainnollistetaan miten JavaScript toimii esimerkiksi laskimena, joka summaa kaksi lukua keskenään ja antaa tuloksen suoraan haluttuun elementtiin. JavaScript ei tarvitse palvelinta laskutoiminnon suorittamiseen vaan sen tekee käyttäjän tietokone. Seuraavassa luvussa sama tehdään PHP:lla, joka puolestaan tarvitsee laskutoimitukseen palvelimen.

```
<form>
   Luku 1 <input type=text id="L1">
   Luku 2 <input type=text id="L2">
   <input type=button value="Summaa"
      onclick="summaaLuvut();">
</form>

<div id="tulos"></div>

<script type="text/javascript">

function summaaLuvut() {
   var luku1 = document.getElementById('L1').value;
   var luku2 = document.getElementById('L2').value;
   var summa = parseFloat(luku1) + parseFloat(luku2);

   document.getElementById('tulos').innerHTML =
      'Summa on ' + summa;
}

</script>
```

Listaus 3.3-1. JavaScriptillä toteutettu laskin, joka summaa kaksi lukua keskenään ja tulostaa vastauksen suoraan haluttuun elementtiin ilman palvelimen apua.

3.4 PHP

PHP on JavaScriptistä poiketen palvelinpuolen ohjelmointikieli, jota myös käytetään sivujen dynaamisuuden luomiseen omalla tavallaan. Se on komentosarjakieli, jossa koodi tulkitaan ensin palvelimella ja lähetetään käyttäjän selaimelle HTML -merkkaukseksi tulkattuna. Se voidaan sisäistää HTML:n sekaan. PHP -lähdekoodia ei voida sellaisenaan lukea asiakaspäässä. (Php.netin www-sivut 2012)

PHP sisältää laajan luokkakirjaston, josta löytyy toimintoja useille eri alueille kuten merkkijonojen ja päivämäärien käsittelyyn, useiden eri tiedostoformaattien käsittelyyn ja erilaisille tietokantajärjestelmille. Hyvä puoli on myös sen selainriippumattomuus. Koska koodi tulkitaan jo palvelinpäässä ja siitä tuloksena aikaansaadaan HTML -koodia, se toimii selaimissa aina samoin.

Huono puoli on, että toisin kuin JavaScriptillä, pelkällä PHP -koodilla varustettu dynaaminen sivu joudutaan lataamaan kokonaan uudelleen kun jokin toiminto edellyttää pientäkin muutosta sivun sisältöön. Sellaista sivua on hidas käyttää ja se kuormittaa verkkoa. Esimerkkinä voi ajatella tuhannen tuotteen luetteloa. Kaikista tuotteista on kuva, tuoteseloste ja hinta näkyvillä ja tarvitaan vain yhden tuotteen hinnanmuutos.

Vaatimuksena työssä on, ettei sivua ladata uudelleen kun sivun sisältöön vaikuttava toiminto tehdään, jotta käyttöliittymä saataisiin tuntumaan työpöytäsovellukselta. JavaScript ei voi keskustella verkossa sijaitsevan tietokannan kanssa, mutta PHP voi. Näin ollen PHP on yhdistettävä JavaScriptin kanssa ja käytettävä molemmista hyödyksi niiden hyvät ominaisuudet.

Seuraava listaus havainnollistaa, miten PHP toimii laskimena, joka summaa luvut keskenään ja antaa tuloksen haluttuun elementtiin.

```php
<form method=post>
    Luku 1 <input type=text name="L1">
    Luku 2 <input type=text name="L2">
    <input type=submit value="Summaa">
</form>

<div id="tulos"><?php summaaLuvut(); ?></div>

<?php

function summaaLuvut() {
    $luku1 = $_POST['L1'];
    $luku2 = $_POST['L2'];
    $summa = $luku1 + $luku2;

    echo 'Summa on '.$summa;
}

?>
```

Listaus 3.4-1. PHP:llä toteutettu laskin, joka summaa kaksi lukua keskenään ja tulostaa vastauksen haluttuun elementtiin.

Toiminta on lähes sama kuin edellisen luvun JavaScript -laskimessa. Pienen toiminnallisen eron lisäksi löytyy se suuri ero, että PHP -versio lähettää käyttäjän syöttämät luvut selaimelta palvelimelle samalla kun se käynnistää uudelleen itsensä saamiensa lukujen kanssa. Sovellus laskee summan palvelimella ja koko sivu lasketun tuloksen kanssa lähetetään selaimelle.

Tässä tapauksessa PHP -versio ei tosin ole juuri JavaScript-versiota hitaampi. Nopeuseroihin vaikuttavat aina toimintojen monimutkaisuus ja tietomäärien erot. Tässä tapauksessa erot ovat pienet, kyse on vain kymmenien tavujen erosta.

Seuraavana on vielä havainnollistettu, miten PHP:n skripti ei näy selaimessa, koska skripti on tulkattu HTML -koodiksi jo palvelimelta lähtiessään.

```
<form method=post>
        Luku 1 <input type=text name="L1">
        Luku 2 <input type=text name="L2">
        <input type=submit value="Summaa">
</form>

<div id="tulos">Summa on 8</div>
```

Listaus 3.4-2. Selaimen näkemä lähdekoodi PHP -laskimesta kun laskutoimitus on tehty.

3.5 MySQL

MySQL (My Structured Query Language) on maailman käytetyin avoimen lähdekoodin relaatiotietokantaohjelmisto. Se on alun perin suomalaisen Michael ”Monty” Wideniuksen ja ruotsalaisen David Axmarkin kehittämä ja itse asiassa tämä SQL-järjestelmä on nimetty Michael Wideniuksen My -nimisen tyttären mukaan. (Wikipedian www-sivut 2012)

MySQL on GPL -lisenssin alainen, joten sitä saa käyttää vapaasti, mutta ei levittää kaupallisesti. Merkittäviä MySQL -tietokannan käyttäjiä ovat mm. Wikipedia, Google ja Yahoo!.

MySQL käy monille eri käyttöjärjestelmille. Niihin sisältyvät useat kaupalliset ja vapaat Unixit, eri Windows -versiot ja Linuxit. MySQL:n suorituskyky on huippuluokkaa ja sitä tukee useat eri ohjelmointi- ja skriptikielet. MySQL onkin yleinen toteutusalusta dynaamisille web-sovelluksille. PHP ja MySQL on ehkä yleisimmin toisiinsa liitettyinä www-maailmassa. (MySQL:n www-sivut 2012)

PHP:llä MySQL:n käyttäminen on helppoa. Tässä on esimerkki tietokantayhteyden avaamisesta ja tietokannan valitsemisesta.

```
3   // Tietokantayhteyden avaaminen
4   $connection = mysql_connect("server.fi", "LoginName", "Password")
5     or die("Database connection failure: ".mysql_error());
6
7   // Tietokannan valitseminen
8   mysql_select_db("database", $connection)
9     or die("Database selection failure: ".mysql_error());
```

Listaus 3.5-1. Tietokantayhteyden avaaminen ja tietokannan valinta.

Tietokantakyselyihin ja kaikkiin attribuutteihin ja tietotyyppeihin löytyy tarkat selitykset ja MySQL:n kotisivuilta.

4 AJAX TEKNIIKKA

4.1 Yleiskuvaus

Tänä päivänä netissä on oltava kuvaa, ääntä, videota ja interaktiivisia dynaamisia sovelluksia, kuten esimerkiksi pelejä ja muita ohjelmia. Dynaamisuus web-sivuihin on lisääntynyt, koska on alettu vaatia työpöytäsovellusten kaltaista käyttömukavuutta. Ohjelmistojen asennuspaketeista on haluttu myös eroon, jos kyse on verkkoon liittyvistä ohjelmista. On järkevämpää asentaa sovellukset kerran verkkoon kuin asentaa ne lukuisille asiakaskoneille. Ne ovat verkossa käytettävissä joka laitteella, josta on yhteys nettiin, esimerkiksi älypuhelimella. (Asleson & Scutta 2007, 1-6)

Palvelinpuolen ohjelmointikielillä saadaan aikaiseksi dynaamisuutta ja interaktiivisuutta. Yleisesti käytössä on esimerkiksi Perlin tai PHP:n kaltaisia kieliä. Näillä on kuitenkin se yhteinen ongelma, että kun joku osa sivun sisällöstä halutaan muuttaa, koko sivu joudutaan lataamaan uudelleen. Ajaxin tarkoituksena on verkkoliikenteen minimointi ja maksimoida sivun toimintanopeus. Ajax-toiminto reagoi käyttäjän toimenpiteeseen suoraan ilman sivun uudelleen lataamista. Sen avulla voidaan päivittää sivulta vain tietty osio, joten tietoa liikkuu verkon yli murto-osa koko sivun uudelleenlataamiseen nähden.

Ajax on järkevä ratkaisu verkkosovellusten ohjelmointiin, koska se käyttää hyväkseen jo olemassa olevia web-kehitystekniikoita. Näin ei tarvitse opetella uusia ohjelmointikieliä ja selaimeen asennettavat liitännäiset saa unohtaa. Ajax on muuten täysin selainriippumaton tekniikka, mutta se vaatii selaimelta JavaScript-tuen.

4.2 Vahvuudet ja heikkoudet

Ajax-tekniikan vahvuutena on verkon kuormittamisen vähyys ja tästä seuraa samalla sivun nopeampi toiminta. Vaikka www-tekniikkaa ei ole suunniteltu monimutkaisille sovelluksille, Ajaxia käyttämällä sivut saadaan kuitenkin toimimaan hyvin dynaamisina ja interaktiivisina, kuten työpöytäsovellukset. Ajax koostuu joukosta vanhoja tekniikkoja, joiden osaajia on paljon. Siksi sillä on kova suosio. Se on myös selainriippumaton.

Heikkouksiakin Ajaxissa on, eikä sitä pidä käyttää ihan joka tilanteessa. Koska Ajax-toimintojen aiheuttamat muutokset sivun sisältöön eivät välity osoiteriville, sivua ei voi sellaisenaan lisätä suosikeihin tai kopioida suoraa osoitetta toiselle. Koska sivulla olevat dynaamiset Ajax-osiot eivät aidosti kuulu sivun sisältöön, hakukoneet eivät löydä sivua niiden osioiden perusteella. Ajax myös vaatii JavaScript-tuen. Toisinaan voi käydä niin, että asynkroninen Ajax-kutsu voi palvelimen ruuhkasta riippuen palauttaa vastauksen viiveellä.

4.3 XMLHttpRequest -objekti – Ajaxin ydin

4.3.1 Objektin historia lyhyesti

Ajaxin ytimenä toimii XMLHttpRequest-objekti (myöhemmin XHR). Se on JavaScript -funktio, joka on ensimmäisenä sisällytetty Internet Explorer 5 -selaimeen ActiveX -komponenttina vuonna 1999. Laajaa käyttöä sillä ei kuitenkaan ollut ennen kuin tuki kehitettiin myös muihin yleisimpiin selaimiin, kuten Mozillaan vuonna 2002 ja Safariin vuonna 2004. Internet Explorer 7.0 ilmestyi 2006. Siihen on sisällytetty aito XHR -objekti ja ActiveX -komponentista on päästy tältä osin eroon. (Asleson & Scutta 2007, 25)

4.3.2 XMLHttpRequest -objektin luominen

Microsoft tukee edelleen Windows XP:tä, jonka mukana tulee Internet Explorer 6.0 ja jossa tukea XHR -objektille ei ole. Jos web-sivulla halutaan ottaa huomioon Internet Explorerin vanhat versiot, JavaScript koodin pitää tarkistaa tukeeko selain suoraan XHR -objektia vai ei. Seuraavana on yksinkertainen esimerkki objektin selainriippumattomaan luomiseen. Internet Explorerin eri versioita varten on olemassa vielä yksityiskohtaisempia tarkennuksia, mutta tästä selviää periaate.

```
1   var xmlHttp;
2
3   function createXMLHttpRequest() {
4       if (window.XMLHttpRequest) {
5           xmlHttp = new XMLHttpRequest();
6       }
7       else if (window.ActiveXObject) {
8           xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
9       }
10  }
```

Listaus 4.3.2-1. XHR -objektin luominen selainriippumattomalla tavalla.

Rivi 1: Luodaan globaali muuttuja xmlHttp, jolla myöhemmin tullaan viittaamaan varsinaiseen XHR -olioon. Rivi 3: Luodaan JavaScript -funktio tulevan ohjelmoinnin helpottamiseksi. Rivi 4: Kokeillaan, löytyykö selaimesta tuki XHR -objektille suoraan. Jos löytyy, objekti toteutetaan normaalina JavaScript -oliona. Rivi 7: Jos XHR -tukea ei ollut, kokeillaan tukeeko selain ActiveX -komponentteja. Jos tukee, olio luodaan ActiveX:n avulla. (Asleson & Scutta 2007, 26)

Edellä esitelty objektin luominen on ainoa selainkohtainen eroavaisuus Ajax-tekniikassa. Luomalla objekti tällä tavoin, XHR:n ominaisuuksia voidaan ohjelmakoodissa käyttää selainriippumattomasti ja tämä luomistapa nopeuttaa ohjelmistojen kehittämistä. (Asleson & Scutta 2007, 26)

4.3.3 Kyselyn suorittaminen

Seuraavana on esitetty kooditasolla miten tietokantakysely tehdään Ajax-tekniikkaa hyväksikäyttäen yksinkertaisesti. Toiminta selvitetään siinä järjestyksessä kuin koodi etenee.

Esimerkin sovelluksessa käytetään muutamaa erillistä tiedostoa, joista Ajax-tekniikan kannalta tärkeimmät esitellään.

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "
    http://www.w3.org/TR/xhtml1/DTD/chtml1-strict.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4     <title>Ajax request</title>
5     <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
6     <script type="text/javascript" src="functions.js"></script>
7   </head>
8
9   <body>
10
11    <input type=button value="Tee tietokantahaku" onclick="makeRequest();">
12
13  </body>
14  </html>
```

Tiedosto 1: request.html

Tämä on normaali HTML -sivu, jossa on vain yksi painike. Sivun HEAD -osiossa on määritelty, että JavaScript -funktiot löytyvät functions.js -tiedostosta. Painiketta klikattaessa, kutsutaan makeRequest -funktiota, joka luetaan functions.js -tiedostosta.

```
var xhr = false;

// luodaan selainriippumaton xhr olio Ajax-pyynnöille
function createXHR() {
  if (window.XMLHttpRequest)
    xhr = new XMLHttpRequest();
  else
    if (window.ActiveXObject)
      xhr = new ActiveXObject("Microsoft.XMLHTTP");
}

// toimenpide
function makeRequest() {
    createXHR();
    xhr.onreadystatechange = actionDone;
    xhr.open("GET", "action.php", true);
    xhr.send(null);
}

// tässä alert-boksiin tulostetaan joko virhe tai
// palvelimen vastaus pyyntöön
function actionDone() {
    if (xhr.readyState == 4 && xhr.status != 200) {
        alert('Error. Status is ' + xhr.status);
    }
    else if (xhr.readyState == 4 && xhr.status == 200) {
        alert('Server response is ' + xhr.responseText);
    }
}
```

Tiedosto 2: functions.js

Tiedosto sisältää kaikki tarvittavat funktiot. Ensimmäisenä luodaan selainriippumaton XMLHttpRequest -olio, xhr. MakeRequest -funktion skripti luo ensin edellä mainitun olion. Seuraavaksi se käskee, että aina kun olion sisäinen tila muuttuu, kutsutaan actionDone -funktiota. Sen jälkeen avataan action.php -tiedosto, jolta jäädään odottamaan vastausta. Koska pyyntö suoritettiin nyt asynkronisena, vastauksen odottaminen ei jätä web-sivua jumiin käsittelyn ajaksi, esimerkiksi palvelimen ruuhkan takia tai jos pyyntö aiheuttaisi pidemmän prosessin palvelimella. Sivulla voitaisiin tällä välillä tehdä muita toimintoja. Vastaus tulee palvelimelta, kun toiminto on suoritettu.

Joka kerta kun actionDone -funktiota kutsutaan, Ajax-pyynnön tila tarkistetaan. Kun on saavutettu viimeinen tila (4) eli pyyntö on suoritettu loppuun, tarkistetaan vielä onko sen status ”ok” (200). Silloin kaikki olisi kunnossa ja huomiolaattaan tulostetaan palvelimelta saatu vastaus. Jos se on joku muu, tulostetaan käyttäjälle huomiolaattaan virheilmoitus ja status, johon kysely päätyi.

```php
<?php

// avataan tietokantayhteys ja valitaan tietokanta
// erillisellä tiedostolla
// @ -merkki piilottaa mahdolliset PHP:n virheilmoitukset käyttäjältä
@include("openDatabase.php");

// haetaan table-nimisestä tietokantataulusta kaikkirivit
$rs = @mysql_query("SELECT * FROM table");
// ...ja niiden lukumäärä
$nr = @mysql_num_rows($rs);

// tulos lähtee nyt takasin sinne mistä sitä pyydettiin
echo $nr;

?>
```

Tiedosto 3: action.php

Yllä olevan koodin kommentointi selittää miten skripti hakee tiedon ulkopuolisesta tietokannasta. Näyttää siltä, että tämä ohjelma tulostaa tietokantakyselyn antaman vastauksen suoraan käyttäjän selaimeen. Sen se itse asiassa tekeekin, jos tämä tiedosto ajetaan suoraan selaimen osoiteriviltä, eli silloin selain pyytäisi vastausta. Mutta nyt pyyntö tulee Ajaxilta, joten ohjelman tulostama sisältö lähetetään sinne.

4.3.4 Objektin turvallisuus

”XMLHttpRequest-objekti toimii selaimen turvallisen ”hiekkalaatikon” sisällä, jolloin jokaisen XMLHttpRequest -objektin pyytämän resurssin on sijaittava sama verkkotunnuksen (domain) sisällä, josta skriptin suoritus on aloitettu. Tämä turvallisuusrajoite estää XMLHttpRequest -objektia vastaamasta alkuperäisen verkkotunnuksen ulkopuolisiin pyyntöihin.” (Asleson & Scutta 2007, 37)

Tämän turvallisuusrajoitteen vahvuus on selainkohtainen. Internet Explorer antaa vain varoituksen, mutta silti käyttäjä voi halutessaan jatkaa suorittamista. Mozilla Firefox puolestaan keskeyttää suorituksen kokonaan ja antaa virheilmoituksen. Firefoxilla voidaan JavaScriptiä käyttäen ohittaa tarvittaessa kyseinen esto, mutta se ei tietysti ole suotavaa. (Asleson & Scutta 2007, 37-38)

5 JÄRJESTELMÄN TOTEUTUS

Saatavilla ei ollut valmista järjestelmää malliksi, joten työ aloitettiin tyhjästä ja kaikki suunniteltiin itse alusta asti. Jotta kuvaston hallintajärjestelmä voisi toimia Ajax-toiminnoilla, selaimen on tuettava JavaScriptiä. Ja jotta käyttäjätodennus on mahdollinen, käyttäjän on sallittava evästeet.

5.1 Käyttäjätodennus ja HTTP -istunnot

Järjestelmän ylläpitokäyttöliittymään kirjaudutaan henkilökohtaisin tunnuksin erilliseltä kirjautumissivulta. Kun käyttäjä on onnistuneesti tunnistettu, ohjelma käynnistää HTTP -istunnon, joka saa yksilöllisen SID -tunnuksen (Session ID). Samalla luodaan istunto- ja käyttäjäkohtaiset evästeet, joihin tulee tietoa mm. kirjautumisen onnistumisesta. Istunnon käynnistäminen mahdollistaa siihen tarkoitettujen muuttujien käyttämisen. Niihin lisätään käyttäjäkohtaisia hallintajärjestelmän edellyttämiä asetuksia ja tämän jälkeen käyttäjä päästetään jatkamaan hallintasivulle.

Hallintasivun taustalla pyörii koko ajan erillinen ohjelmakoodi, joka tutkii ovatko kaikki istuntoon ja evästeihin sisällytetyt tiedot oikein vai löytyykö niistä virheitä. Virheen löytäessään tämä taustaohjelma ei erittele virheitä käyttäjälle, vaan tuhoaa kaiken kirjautumisen yhteydessä luodun tiedon, eli istuntomuuttujat ja evästeet. Siitä käyttäjä ohjataan kirjautumissivulle.

Jos järjestelmän vaatimia istuntotietoja tai evästeitä ei ole, eli ei olla kirjautuneena, käyttäjä ohjataan aina kirjautumissivulle. Näin käy myös siinä tapauksessa, että yritetään käynnistää mikä tahansa PHP -tiedosto ylläpitoliittymän sijainnista. Mahdollisuus päästä sisälle järjestelmään väärin perustein on olematon.

HTML -lomaketietojen lähetys on suojaamatonta normaalilla HTTP -yhteydellä ja tässä kohtaa on käyttäjätunnuksien varastamisen mahdollisuus, koska tunnus ja salasana lähetetään salaamattomana. SSL -sertifikaattia ei ollut, eikä suojattua yhteyttä tästä syystä voitu käyttää. Sen hankinta ja muu suojattuun yhteyteen liittyvä määrittely jäi työn tilaajayritykselle.

Käyttäjien tunnukset ja salasanat ovat tallennettuina tietokantaan salasanat kryptattuina MD5 -salaustekniikalla, joten niitä ei näe selväkielisenä edes tietokannasta. MD5 -salaustekniikka on yksisuuntainen, joten jos kryptatun salasanan haluaa selväkieliseksi, se pitää kokeilemalla selvittää. Tämä järjestelmä toimii niin, että se vertaa käyttäjän syöttämää salasanaa MD5 -salattuna tietokannassa olevaan ja sen perusteella tekee johtopäätöksensä.

Jos joku käyttäjistä unohtaa salasanansa, korkeimman käyttöoikeustason käyttäjät voivat käyttöliittymästä nappia painamalla antaa käyttäjälle uuden salasanan. Järjestelmä arpoo automaattisesti uuden, jonka se lähettää siihen sähköpostiosoitteeseen, joka on merkitty kyseisen käyttäjätilin tietoihin. Salasanaa ei siis näe kukaan muu kuin se henkilö, jonka sähköpostiin se tulee.

5.2 Käyttöliittymä

5.2.1 Käyttöoikeustasot

Käyttöoikeustasoja on kolme. Alimman tason käyttäjillä on oikeus lisätä tuotteita tietoineen ja kuvineen järjestelmään sekä poistaa niitä. Keskitason käyttäjät voivat näiden toimintojen lisäksi hallinnoida myös kategoriointia. Korkeimman tason käyttäjillä on käytettävissään kaikki mahdolliset toiminnot. Nämä käyttäjät voivat esimerkiksi asettaa alempien tasojen käyttäjille eston päästä ylläpitoliittymään. He voivat lisätä käyttäjiä järjestelmään ja muuttaa heidän käyttöoikeustasojaan. Käyttäjien määrää ei ole rajoitettu millään käyttöoikeustasolla.

5.2.2 Oman tilin hallinta

Jokainen käyttäjä voi muokata omia tietojaan. Käyttöliittymän Oma tili -välilehdeltä on nähtävissä käyttäjän oikea nimi, tilin aktivointipäivämäärä ja salasanan ikä päivissä. Näitä tietoja ei voida muuttaa. Muutettavia tietoja ovat sen sijaan käyttäjätunnus, sähköpostiosoite ja puhelinnumero. Nykyisen ja uuden salasanan kentät ovat tyhjinä.

Vaikka käyttäjä on jo tunnistettu kun hän on kirjautunut järjestelmään, hänen on silti kirjoitettava salasanakenttään nykyinen salasanansa oikein, jos haluaa muuttaa tietojaan. Uutta salasanaa ei tarvitse antaa, jos nykyinen on tarpeeksi tuore. Jos uusi annetaan, se ei voi olla sama kuin edellinen ja se on oltava tiettyjen kriteerien mukainen.

5.2.3 Kategoria- ja tuotehallinta

Tämän välilehden sisältö on kolmiosainen. Ensimmäisessä osiossa on listattuna kaikki kategoriat puumaisena luettelona. Samaan osioon kuuluu toiminnot kategorioiden lisäykseen, poistoon ja nimen vaihtoon. Nämä toiminnot puuttuvat alimman käyttöoikeustason käyttäjiltä.

Toisessa osiossa on listattuna tuotteet, jotka sisältyvät ensimmäisestä osiosta valittuun kategoriaan. Samassa yhteydessä on toiminnot tuotteen lisäämiseen ja poistoon.

Kolmas osio on tuotetietoja varten. Se sisältää toisesta osiosta valitun tuotteen tiedot ja näyttää kaikki siihen linkitetyt kuvat pieninä esikatselukuvina. Tässä osiossa voidaan muuttaa tuotteen tietoja sekä lisätä tai poistaa tuotteeseen liittyviä kuvia. Kuvia ei tarvitse ladata eikä niiden määrää myöskään ole rajoitettu.

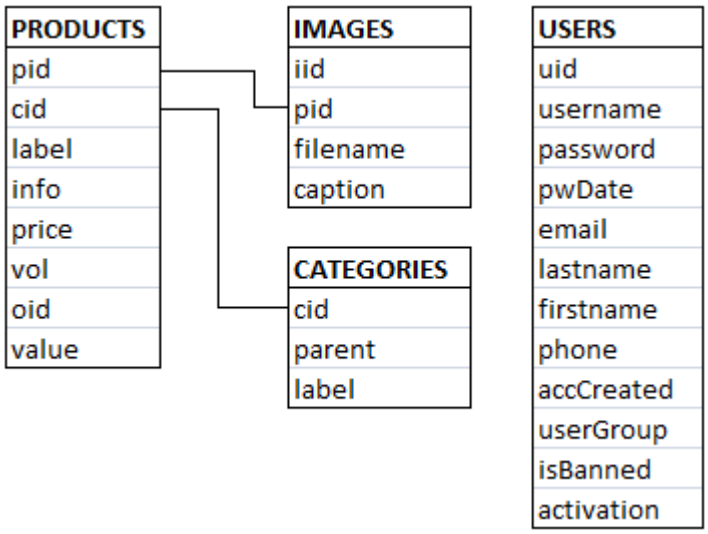
5.2.4 Käyttäjätilien hallinta

Tämä välilehti on käytössä vain korkeimman käyttöoikeustason käyttäjillä. Näkyvillä on luettelo kaikista käyttäjätileistä. Yksi rivi sisältää aina yhden käyttäjätilin. Siitä näkee käyttäjän koko nimen, sähköpostiosoitteen, puhelinnumeron, käyttöoikeustason ja tilan siitä, onko käyttäjä estetty järjestelmästä. Näitä tietoja voidaan muokata suoraan tästä rivistä ja tallentaa. Rivillä on myös painikkeet käyttäjätilin poistoon ja uuden salasanan antamiseen.

Samalta välilehdeltä voi lisätä käyttäjätilin. Tyhjiin kenttiin annetaan vähintään käyttäjän nimi ja sähköpostiosoite. Puhelinnumeron antaminen on vapaaehtoista.

5.3 Tietokantarakenne

Tietokantarakenne on hyvin yksinkertainen, ohjelmiston ollessa vasta pohja jollekin tulevalle. Se piti tässä vaiheessa jättää avoimeksi muutoksille, joten sitä ei alettu virittämään tietynlaiseksi. Hallintajärjestelmän prototyyppi toimii vaatimusten edellyttämällä tavalla.



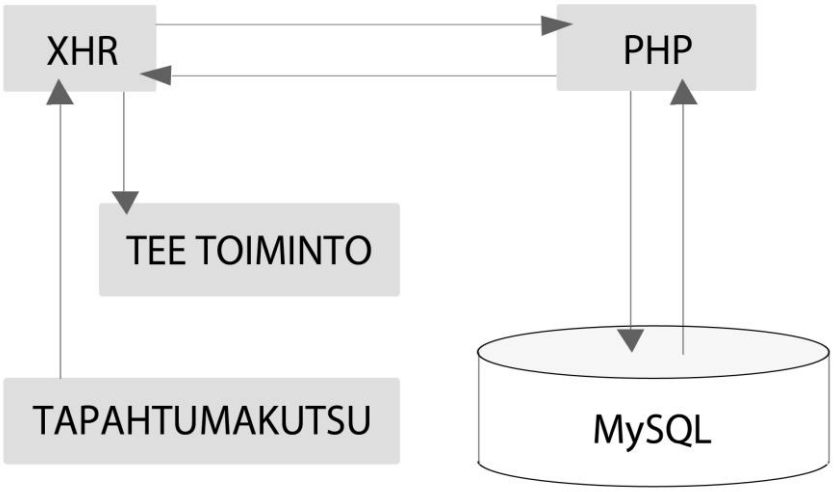
Kuva 5.3-1. Tietokantataulut.

Vielä tällaisella rakenteella ei voida antaa mahdollisuuksia muuttaa tai lisätä tuotteiden optioita käyttöliittymästä käsin, koska samalla annettaisiin mahdollisuus tietokantataulun rakenteen muuttamiseen. Optioita eli tuotteen ominaisuuksia on tällä rakenteella muokattava suoraan tietokannasta.

Ohjelmistoa tullaan jatkossa siis kehittämään. Tietokantarakenteeseen tullaan tekemään ratkaisevia muutoksia. Erityyppisille tuotteille pitää pystyä antamaan erilaiset ominaisuudet, ja ominaisuuksille pitää voida antaa erityyppisiä arvoja. Sillä ratkaisulla web-kuvasto eli julkinen käyttöliittymä saadaan hakuominaisuuksiltaan järkevämmäksi. Tämä työ sisälsi vain vaatimuksen kategorioidun rakenteen luomisesta, eikä aika riittänyt yksityiskohtaisempaan virittämiseen tällä erää.

5.4 Järjestelmän sisäinen tiedonsiirto

Seuraavana on esitelty lohkokaaviona, miten tieto kulkee selaimen ja palvelimen välillä järjestelmän sisällä.



Kuva 5.4-1. Lohkokaavio järjestelmän sisäisestä tiedonsiirrosta.

Otetaan tähän esimerkiksi joku järjestelmässä oleva toiminto, vaikka oman salasanan vaihtaminen ja lähetys omaan sähköpostiin.

Tapahtuma lähtee käyntiin HTML -sivulta esimerkiksi seuraavalla koodilla.

```
<input type=button value="Uusi Salasana" onClick="vaihdaSalasanani();">
```

Luodaan XMLHttpRequest -ilmentymä, avataan yhteys PHP:hen open() -metodilla ja kerrotaan, että halutaan käynnistää vaihdasalasana.php -tiedosto GET -metodilla. Pyyntö tapahtuu lopuksi send() -metodilla, seuraavan listauksen mukaisesti. Action-Done() -funktio antaa virheilmoituksen heti , jollei pyyntö toimi halutulla tavalla.

```
function vaihdaSalasanani() {
  var xhr = false;

  if (window.XMLHttpRequest)
    xhr = new XMLHttpRequest();
  else if (window.ActiveXObject)
    xhr = new ActiveXObject("Microsoft.XMLHTTP");

  var url = "vaihdasalasanani.php";

  xhr.onreadystatechange = actionDone;
  xhr.open("GET",url,true);
  xhr.send(null);
}

function actionDone() {
  if (xhr.readyState == 4 && xhr.status != 200) {
    alert('Error!');
  }
}
```

Listauksessa ei lue käyttäjän tietoja missään vaiheessa, joten niitä ei kukaan voi lähdekoodista nähdä. Kaikki tarvittava käyttäjään liittyvä tieto on tallennettu istuntomuuttujiin.

Seuraava PHP -koodi käynnistyy järjestelmän taustalla.

```
$userId = $_SESSION['userID'];
$email = $_SESSION['userEmail'];
$password = generatePassword(10);
$md5password = md5($password);

$qr = "UPDATE users SET password='$md5password' where id=$userId limit 1";

if (mysql_query($qr)) {
  $message = "Uusi salasanasi on $password";
  $to      = $email;
  $subject = 'Uusi salasanasi';
  $headers = 'From: no-reply@testpage.fi';

  @mail($to, $subject, $message, $headers);
}
```

Se luo satunnaisen salasanan generatePassword -funktiolla, jota ei tässä yhteydessä esitellä. Sen jälkeen se päivittää tietokantaan kyseiselle käyttäjälle MD5-salatun salasanan. Jos tämä onnistuu, ohjelma lähettää selkokielisenä salasanan käyttäjän sähköpostiin.

Tuosta listauksesta puuttuu ilmoituksen antaminen takaisin. Se voisi ilmoittaa paluuviestinä tehtiinkö toiminto onnistuneesti vai tuliko jossain virhe ja mikä virhe.

6 JOHTOPÄÄTÖKSET

Tavoitteena oli opetella luomaan dynaamisempia web-sovelluksia kun aikaisemmin olen toteuttanut pelkän PHP:n ja MySQL:n avulla. Ennen työn aloittamista Ajax-tekniikka ja JavaScript-ohjelmointikieli olivat tuntemattomia. Tämä teki työstä mielenkiintoisen sekä haastavan ja uutta asiaa opittiin paljon. HTML ja CSS olivat jo ennalta tuttuja. PHP ja MySQL sekä niiden keskinäinen toiminta oli opiskeltu perustasolla. Oppimista tapahtui silti niidenkin osalta jälleen. HTTP-istuntotoiminnot tuli opiskeltua perusteellisesti, koska ne olivat tuntemattomia ja tässä työssä erittäin tärkeitä.

Koska järjestelmässä on nyt vain perustoiminnot tuotekuvaston rakentamiseen, niin se ei ole kovin käytännöllinen. Esimerkiksi tuotetta ei voi poistaa, jos siihen liittyy kuvia. Tai kategoriaa ei voi poistaa, jos se sisältää tuotteita. Nämä ongelmat ratkaistaan laittamalla perustoimintoja toimimaan peräkkäin automaattisesti. Perustoiminnothan kaikkeen muokkaamiseen on jo olemassa ja muuta työltä vaadittukaan.

Toinen asia on tietokantarakenteen tuotetaulun yksinkertaisuus. Alkuperäinen tarkoitus oli jättää järjestelmä avoimeksi sen kehittämiselle. Tällä pohjalla luodun kuvaston hakuominaisuuksia ei saada kovin älykkäiksi. Tuotteen selosteesta voidaan kyllä tehdä tietokantahakuja, mutta nopeaa se ei tule olemaan jos tuotteita on paljon. Tulevassa kuvaston julkisessa käyttöliittymässä ei myöskään voi järkevästi jäsennellä tuotteita. Ne täytyy jäsennellä niin kuin ne on kategorioitu. Tuotteiden ominaisuuksia varten pitäisi tehdä oma taulukko ja linkittää se sitten tuotetaulukkoon. Näin erityyppisillä tuotteilla voisi olla erityyppisiä ominaisuuksia ja tuotteiden jäsentely paranisi.

Koko järjestelmän olisi voinut tehdä myös Flash -sovelluksena. Sillä formaatilla on mahdollista tehdä aivan samanlainen sovellus ja myös visuaalisesti parempi helpommin. Flash -sovellukset eivät kuitenkaan toimi ilman selaimeen asennettavaa liitännäistä. Myös Java-applettina päästäisiin samaan, mutta sekin vaatii lisäosien asen-

tamista tietokoneelle. Ajaxille riittää JavaScript-tuki ja se löytyy useimmista selaimista ja siksi työ toteutettiin Ajax-tekniikalla.

Samankaltaisen järjestelmän olisi voinut toteuttaa Drupal tai Jooma -julkaisujärjestelmiin suunnittelemalla sinne oma moduuli. Työn idea olisi kuitenkin hukkunut. Tarkoitus oli, että on tämä on täysin itsenäinen järjestelmä.

Hallintasovelluksen ohjelmakoodi on kirjoitettu niin, että sitä hieman muuttamalla on helppo räätälöidä järjestelmä joksikin aivan muuksi kuin juuri tuotekuvaston hallintasovellukseksi. Muokkaaminen vaatii kuitenkin hieman kokemusta ohjelmoinnista. Järjestelmälle tuli perusteellisen hyvä Ajax-runko. Eli pohjakoodista tuli selkeä ja ulkopuolisen helposti ymmärrettävä.

LÄHTEET

Wikipedian www-sivut. Viitattu 16.5.2012. http://www.wikipedia.org/

Web-opas www-sivut. Viitattu 16.5.2012 http://www.webopas.net

Php-netin www-sivut. Viitattu 29.5.2012 http://www.php.net

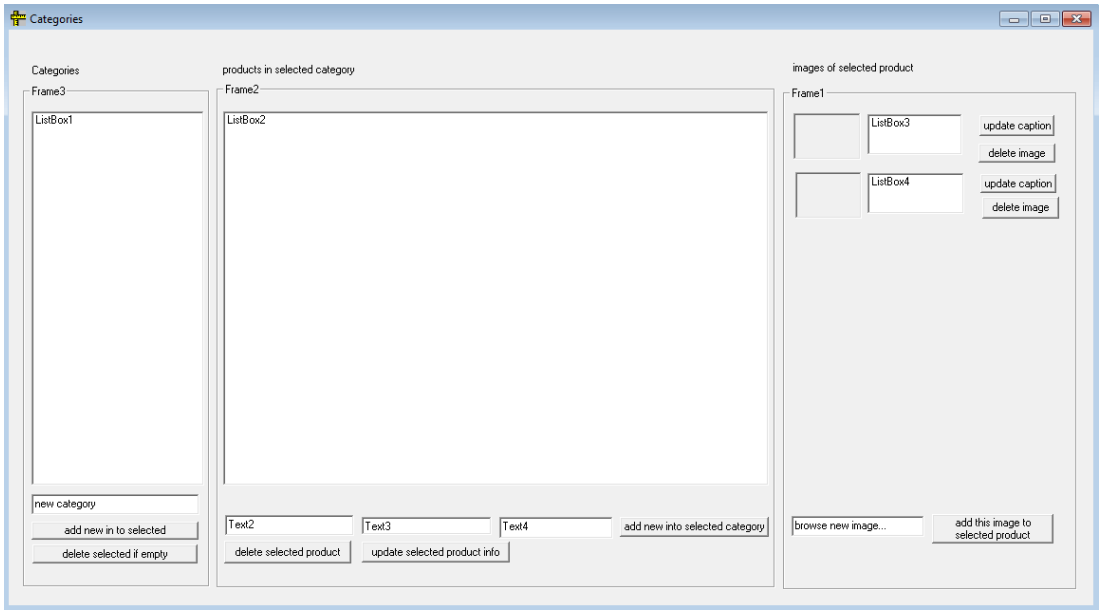
MySQL:n www-sivut. Viitattu 27.5.2012 http://www.mysql.com

Negrino, T., Smith, D. JavaScript: Tehokas Hallinta. Jyväskylä: Gummerus, 2007

Asleson, R., Scutta, N. Ajax: Tehokas Hallinta, Jyväskylä: Gummerus, 2007

LIITE 1: Kuvat ylläpitopaneelin kategoriointi- ja tuotehallintaosiosta.

Suunnitteluvaiheen ulkoasu



Lopullinen ulkoasu